

# Федеральное агентство морского и речного транспорта Федеральное государственное бюджетное образовательное учреждение

**высшего образования**

# «Государственный университет морского и речного флота имени адмирала С.О. Макарова»

**Институт водного транспорта Кафедра: ВСИ**

# ОТЧЕТ

**По лабораторной работе на тему**

# «ФУНКЦИЯ – Создание многофайлового проекта» Дисциплина: Технологии программирования

**Вариант № 11**

# Выполнил:

# Студент группы ИС-22,

**Миронов К.С.**

**8(932)852-07-54**

[**Kirillmironov123@gmail.com**](mailto:Kirillmironov123@gmail.com)

# Проверил: доц. каф. ВСиИ.

**к.г.н., Балса А.Р.**

# г. Санкт-Петербург 2023г.

# 1. ПОСТАНОВКА ЗАДАЧИ ДЛЯ СЕМИНАРА 3

В одномерном массиве, состоящем из n целых элементов:

1) найти значение, количество и координаты минимального по модулю элемента массива;

2) найти сумму модулей элементов массива, расположенных после первого отрицательного элемента. Найти положение отрицательного элемента, после которого найдена сумма модулей;

3) сжать массив, удалив из него все элементы, величина которых находится в интервале [a,b](a–левая граница интервала, b–правая граница (a<b)). Освободившееся в конце массива элементы заполнить нулями

# АНАЛИЗ

**Задача 1.** Найти значение, количество и координаты минимального по модулю элемента массива

# Не успешные сценарии:

1. **Проверка размера массива**

**Ввод**: **0**

# Вывод: как видим, массив пуст.

**Ответ**: **Задача не может быть выполнена!**

Объяснить пользователю, что массив не может быть пуст.

1. **Минимальных по модулю значений – несколько, с разным знаком**

**Ввод: -3 8 7 3 21 8**

**Вывод:** как видим, минимальное по модулю – число -3 и 3.

**Ответ:** вывести на экран как отрицательное, так и положительное значение.

**Задача 2.** найти сумму модулей элементов массива, расположенных после первого отрицательного элемента. Найти положение отрицательного элемента, после которого найдена сумма модулей;

# Не успешные сценарии:

1. **Проверка размера массива**

# В массиве нет отрицательных элементов

Пусть в массиве 3 элемента

# Ввод: 1 8 5

**Вывод:** как видим, все элементы положительны

**Ответ: Задача не может быть выполнена!**

Объяснение пользователю, что в массиве для выполнения задачи нужен хотя бы один отрицательный элемент

# Первый отрицательный элемент – последний элемент в массиве

Пусть в массиве 4 элемента:

# Ввод: 3 3 3 -3

**Вывод:** как мы видим, первый отрицательный элемент - последний элемент массива.

**Ответ:** Сумма не может быть найдена, т.к. сумма – результат сложения как минимум 2 операндов. Объяснить пользователю, что сумма не может быть найдена.

# После первого отрицательного элемента < 2 элементов

Пусть в массиве 5 элементов:

# Ввод: 3 3 3 -3 5

**Вывод:** как мы видим, после первого отрицательного элемента всего 1 элемент.

**Ответ:** Сумма не может быть найдена, т.к. сумма – результат сложения как минимум 2 операндов. Объяснить пользователю, что сумма не может быть найдена.

**Задача 3.** Сжать массив, удалив из него все элементы, величина которых находится в интервале [a,b](a–левая граница интервала, b–правая граница (a<b)). Освободившееся в конце массива элементы заполнить нулями

**Ввод:** интервал [a, b].

**Вывод:** если a>=b, то массив не будет отсортирован.

**Ответ:** объяснить пользователю, что a должно быть меньше b, иначе сортировка невозможна.

# 2. ПОСТАНОВКА ЗАДАЧИ ДЛЯ СЕМИНАРА 4

Коэффициенты системы линейных уравнений можно представить в виде целочисленной прямоугольной матрице размером NxM:

1) с помощью допустимых преобразований привести систему к треугольному виду (описать алгоритм решения данной задачи);

2) после всех преобразований найти количество и координаты строк, среднее арифметическое элементов которых, меньше заданной величины.

# АНАЛИЗ

**Задача 1.** с помощью допустимых преобразований привести систему треугольному виду (описать алгоритм решения данной задачи);

**Алгоритм приведения к треугольному виду:**   
Будем использовать метод Гаусса (приведение матрицы к треугольному виду путём элементарных преобразований).  
**Элементарными преобразованиями матрицы** называются следующие ее преобразования:  
**I.** Перестановка двух столбцов (строк) матрицы.  
**II.** Умножение всех элементов одного столбца (строки) матрицы на одно и то же число, отличное от нуля.  
**III.** Прибавление к элементам одного столбца (строки) соответствующих элементов другого столбца (строки), умноженных на одно и то же число.

Чтобы привести матрицу к ступенчатому виду, нужно выполнить следующие действия:  
**1.** Начинаем с исходной матрицы.

**2.** Выбираем первый столбец и находим ненулевой элемент в этом столбце.

**3.** Если ненулевой элемент находится не в первой строке, меняем местами эту строку с первой.

**4.** Делим первую строку на её первый элемент, чтобы получить единицу на диагонали.

**5.** Для каждой последующей строки вычитаем из неё первую строку, умноженную на коэффициент такой, чтобы получить ноль на первой позиции этой строки.

**6.** Повторяем шаги 2-5 для каждого следующего столбца, начиная с текущей строки и столбца.

**7.** Повторяем шаги 2-6 для каждой последующей строки.

**8.** В результате получаем треугольную матрицу.

# Не успешные сценарии:

# 1.Проверка размера массива.

# Ввод: (0 0) / (0 1) / (1 0) / (1 2) / (2 1)

**Вывод**: как видим, массив не является прямоугольной матрицей.

**Ответ: Задача не может быть выполнена!**

Объяснение пользователю, что матрица должна быть прямоугольной.

# 2.Проверка деления на ноль.

# Ввод: 1 0 0

# 2 0 0

# 0 3 0

# Вывод: как видим, возникает деление на ноль.

# Ответ: если в процессе приведения к треугольному виду одно из диагональных элементов обнуляется, то при делении на него возникает ошибка деления на ноль.

# Нужно ввести дополнительную проверку деления на ноль.

**Задача 2.** после всех преобразований найти количество и координаты строк, среднее арифметическое элементов которых, меньше заданной величины.

# Не успешные сценарии:

# В матрице нет строк, среднее арифметическое элементов которых меньше заданной величины.

# Ввод: {{1, 0, 0}, {0, 3, 0}, {0, 0, 0}}, -17

# Вывод: как видим, таких строк нет.

# Ответ: Задача не может быть выполнена!

# Нужно объяснить пользователю, что таких строк в матрице нет.

# 3.ПОСТАНОВКА ЗАДАЧИ ДЛЯ СЕМИНАРА 5

# Написать программу, которая считывает текст из файла и выводит на экран только строки, не содержащие двузначные числа.

# АНАЛИЗ

**Неуспешные сценарии:**

# Проверка открытия файла

Файл не существует

**Ответ**: Объяснение пользователю, что в файл не удается открыть.

# Проверка наличия текста в файле

Пусть в файле нет текста.

**Ответ**: Объяснение пользователю, что файл пуст.

1. **В файле нет строк без двузначных чисел**.   
   Пусть в файле запись:  
   абвг22  
   17василий82  
   **Ответ:** Объяснение пользователю, что таких строк в файле нет.
2. **В файле 2 числа идут подряд, но первое число – 0.**

Пусть в файле запись:  
Абвг01  
17василий82  
**Ответ:** Такое число не считается двузначным, ведь в нём есть незначащий ноль.

# ПРИЛОЖЕНИЕ

**Пример проверки размера одномерного массива**

#include <iostream> #include <iomanip>

using namespace std; int main()

{

const int N = 50; //Общий размер массива int n = 0; //Для пользователя

char Ok; //ключ для выхода из бесконечного цикла

int\* A = new int[N]; //"заказываем" в куче места для N-элементов, то есть N\*4 байт

//попросим у пользователя ввести размер массива, не больше N

cout << "Please, enter the size of your array not bigger than " << N << " -> "; cin >> n;

//праверка правильности ввода размера массива while(true)

{

if(n <= 0 || n > N)

{

cout << "It's so sorry! The size is not correct!\n"

<< "If you wish to quit, click 'Y' or any other key to continue -> "; cin >> Ok;

if(Ok == 'y' || Ok == 'Y')

{

}

else

{

}

cout << "Good bye!"; break;

cout << "Please, enter again the size of your array not biger than "

<< N << " -> ";

cin >> n;

}

else

break;

}

//ввод элементов массива

cout << "Enter the array's elements:\n"; for(int i = 0; i < n; i++)

cin >> A[i]; // cin >> \*(A + i); В чем разница?

//вывод элементов массива

cout << "You have entered the next elements:\n"; for(int i = 0; i < n; i++)

cout << setw(5) << A[i]; // cout >> setw(5) >> \*(A + i); В чем разница? cout << "\n\n\t\t\t\*\*\*\*\*\*\*For the moment it's all! Bye!\*\*\*\*\*\*\*\n\n";

//Вопрос: Что будет если мы напишем следующий оператор? Почему?

// for(int i = 0; i < N; i++)

// cout << setw(5) << A[i];

delete[] A; //Освобождаем память

A = NULL; //сбрасываем в нуль указатель. Почему?

//cout << \*A; Что может произойти, если попытаться разыменовать указатель? Почему?

//cout << \*NULL; А если попытаться разыменовать нулевой указатель?

return 0;

}